GROUP 33 – GROUP REPORT

CS2810

Team members

Jake Barr
James Lee
Jiajun Liu
Owen Norman
Angelos Tsoutsas
Eduardo Diaz-Rio
Mohammed Khan
Makhmud Abdulkerimov

Statement of Relative Contribution

	Design, Planning and Coordination	Coding and Testing	Other
Jake Barr	19%	20%	18%
James Lee	19%	20%	15%
Jiajun Liu	-	5%	12%
Owen Norman	19%	15%	10%
Angelos Tsoutsas	5%	10%	12%
Eduardo Diaz-Rio	19%	10%	18%
Mohammed Khan	0%	0%	-
Makhmud Abdulkerimov	19%	20%	15%

These percentages were agreed during an in-person team meeting where all team members were presented and all discussed and agreed on said percentages. Below lies each person's signature to show their agreement to the percentages:

Jake Barr	Jake Barr
James Lee	James Lee
Jiajun Liu	刘家俊
Owen Norman	Owen Norman
Angelos Tsoutsas	Angelos Michail Tsoutsas
Eduardo Diaz-Rio	Eduardo Diaz-Rio Varez
Mohammed Khan	Mohammed Raheem Khan
Makhmud Abdulkerimov	Makhmud Abudlkerimov

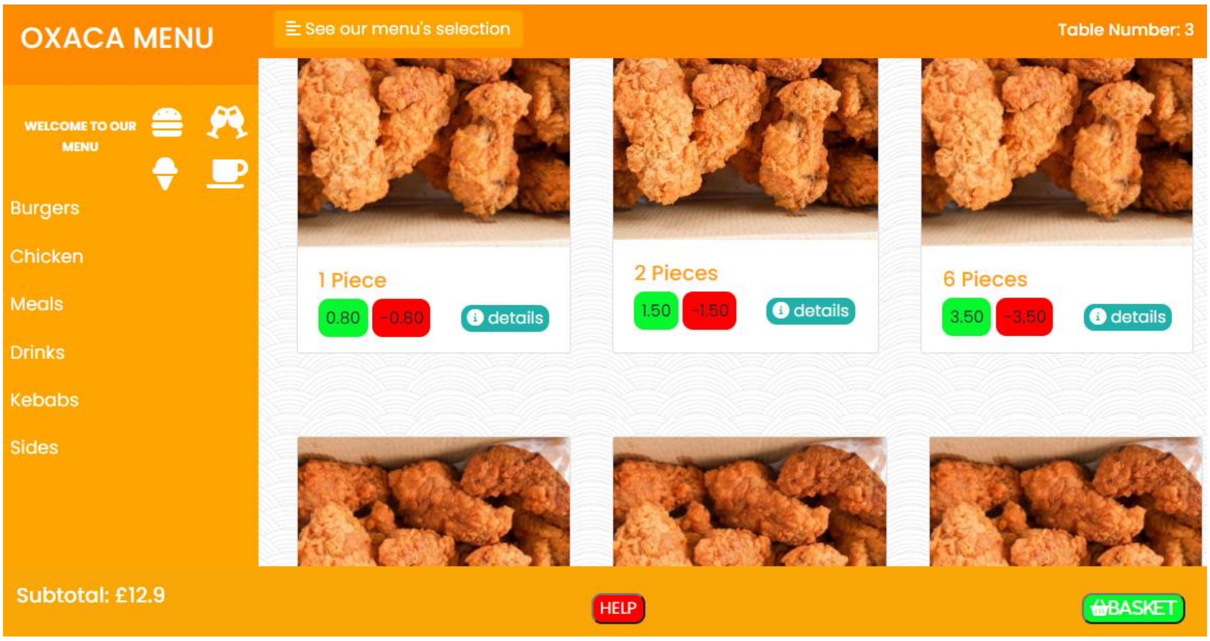
Technical Documentation

Main components

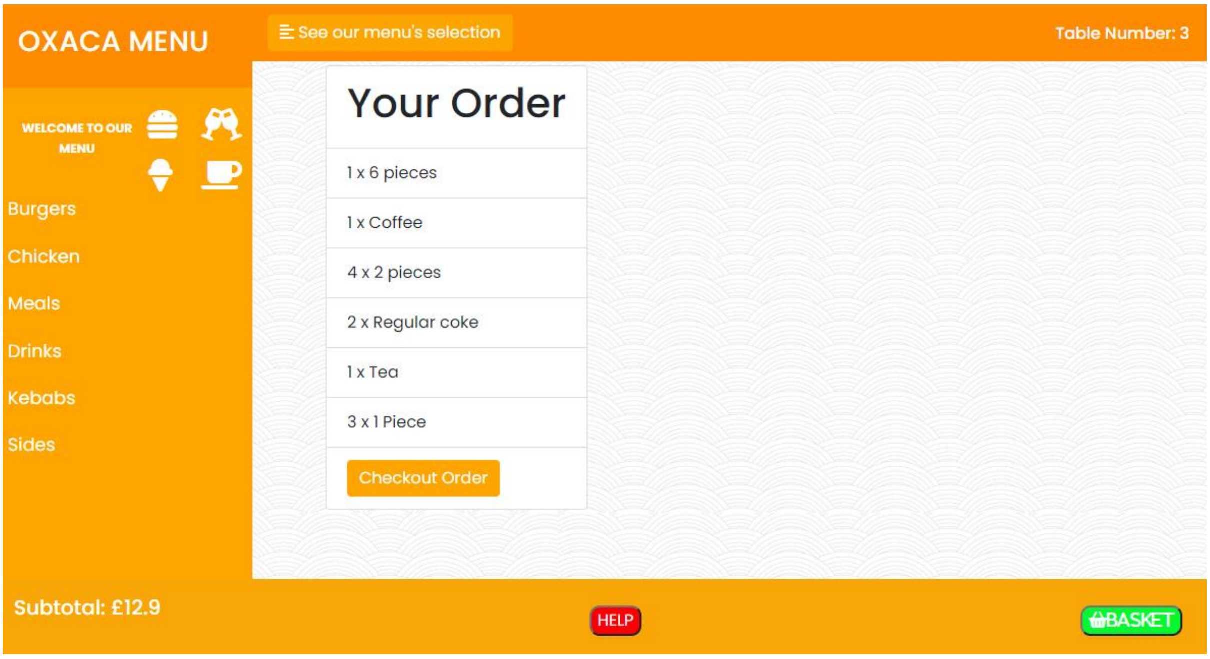
There are 3 sections to our program, the Order UI, the Waiter UI, and the Kitchen UI.

Order UI

This is the section where customers will be ordering from. This can be reached by navigating to here from the intro menu and selecting 'customer'. Here the customer can enter their table number. Once entered the customer will be presented with a variety of submenus, from Burgers to Kebabs and more. The customer will be free to navigate each submenu and add anything they like to their basket and if they make a mistake they can press the red remove button to remove that item. This navigation bar can be collapsed if the customer would like to see more items at a given time. We have made each menu item into a card, which contains a picture of the food, the name, price, and an information button containing any allergic information customers might need. The price also doubles back as a button that when pressed adds said item to the basket. As seen below the submenus are displayed on the left-hand side for easy traversal and if the customer is ever in need of assistance there is a Help button at the bottom where the customer can input their table number and that will alert a staff member. Finally, once the customer has added everything, they can press the basket button in the bottom right corner taking them to the checkout area.



In the checkout area, seen below, the user will be able to see what they have ordered have the ability to go back to order anything else they might want or as said before remove anything. They can also see the price of their order in the bottom left corner. Once they have checked that is everything they want, they will be able to press the checkout order and the order will be sent off for the waiter to review.



Your Order

1 x 6 pieces

1 x Coffee

4 x 2 pieces

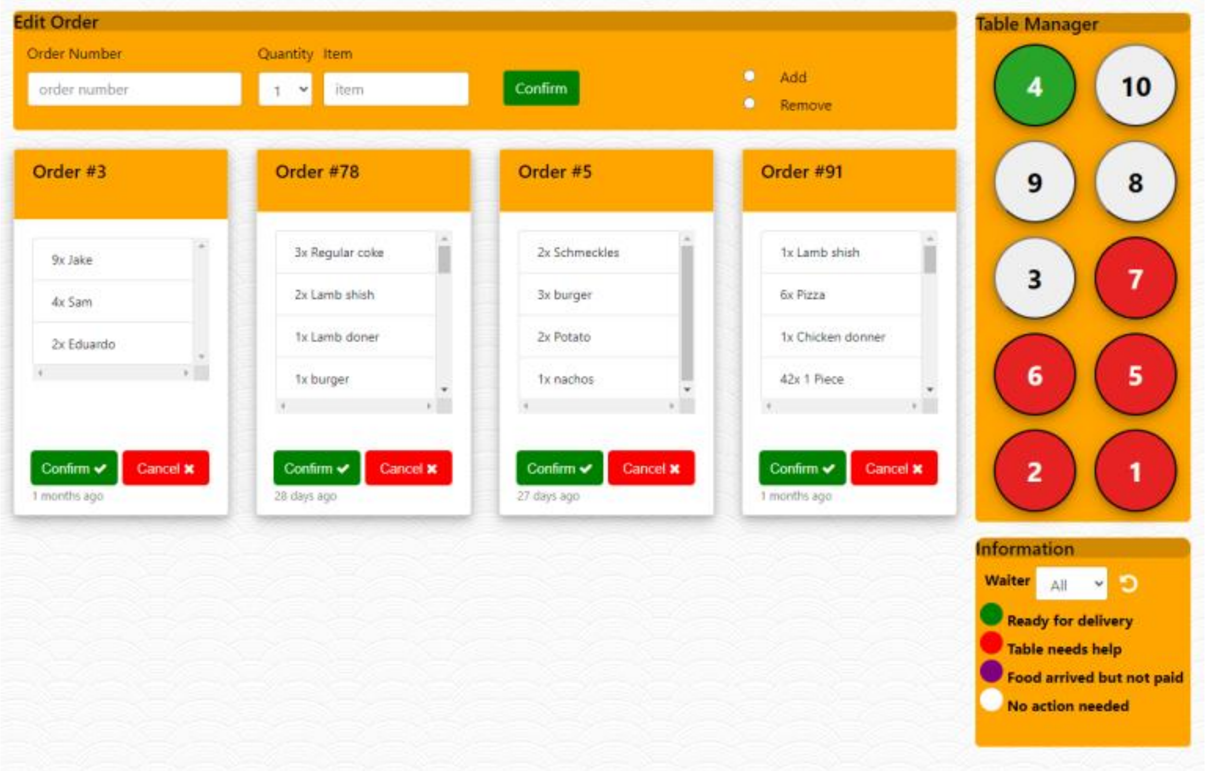
2 x Regular coke

1 x Tea

3 x 1 Piece

Waiter UI

The waiter's role in the system is to interact with orders and manage the tables. The waiter is asked to log in to their account before interaction with the system can begin. This ensures that no customers enter the UI unintentionally. As shown in the image below, the waiter is able to check the table status and see if a table: needs help,is waiting for an order, or if it has received an order and it has yet to pay. Each waiter is able to filter the table manager and display only the tables assigned to him. Each status is represented by a colour as explained in the information box. The waiter can change the status of any table by clicking/touching on the respective circle representing that table. The waiter is also able to interact with orders. After a customer has completed checkout all orders must be first confirmed by the waiter before the preparation of said order can begin. Each order is represented by a card, and it has been given a unique number used for referencing. The waiter has the ability to add and remove items from any given order upon request by the customer. This is done under the “Edit Order” box. The waiter fills in the order number, the item being added or deleted, and the quantity of that item, and selects “Confirm” to complete the edit. When an order is complete, the waiter confirms that order (using the respective button on the order card), essentially “sending” it to the kitchen to begin preparation. Finally, if an order is no longer needed (for example: the customers have left after ordering), the waiter has the ability to cancel an order, and therefore remove it from the system.



Order #3
- 9x Jake
- 4x Sam
- 2x Eduardo

1 months ago

Order #78
- 3x Regular coke
- 2x Lamb shish
- 1x Lamb doner
- 1x burger

28 days ago

Order #5
- 2x Schmeckles
- 3x burger
- 2x Potato
- 1x nachos

27 days ago

Order #91
- 1x Lamb shish
- 6x Pizza
- 1x Chicken donner
- 42x 1 Piece

1 months ago

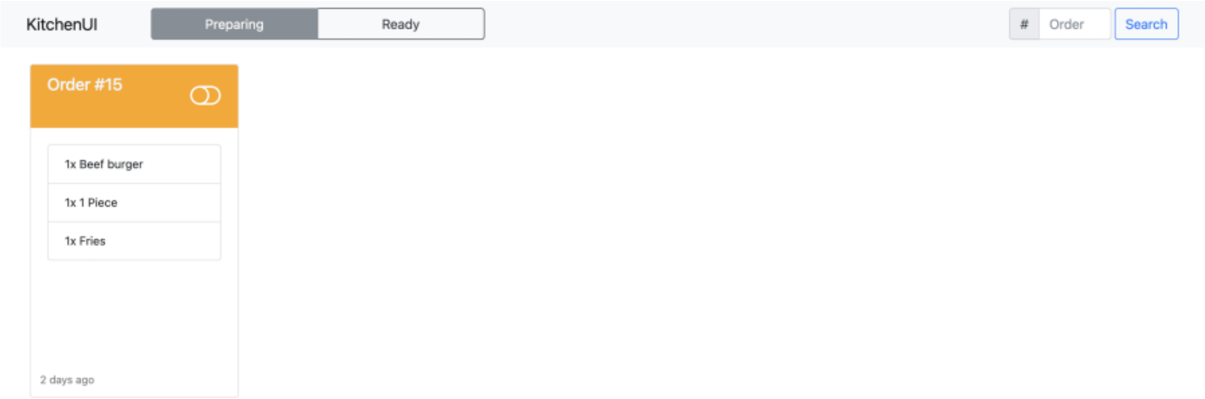
Information

Waiter: All

🟢 Ready for delivery
🔴 Table needs help
🟣 Food arrived but not paid
⚪ No action needed

Kitchen UI

The main use of the Kitchen UI is to interact with incoming orders and to separate them depending on if they are: received, being prepared, ready for delivery. The head chef will be the person interacting with the program as he will be the one telling people what to do in terms of orders. To make it as easy and user-friendly as possible, given the hectic scene of a kitchen, we made it so that the orders will be organised in cards with a button at the bottom of the card that will change the status of the order inside the database when pressed. To keep better track of the orders we separated them into statuses Preparing and Ready this is done to make it easier for the staff to know what needs making and what orders the waiters can take. We also added a search bar so if a customer asks for an update on their order or something went wrong they can find it as fast as possible.



Order #15

1x Beef burger

1x 1 Piece

1x Fries

2 days ago

Packages

Welcome package

This is the first package the user interacts with when the app is run. It is implemented by the following files:

- *introUI.html* formats the way in which the webpage is displayed. It also sets the table number for the order.
- *introUIStyle.css* provides the rules for styling the webpage.
- *vanilla-tilt.js* provides an elegant and user-friendly webpage through the addition of smooth 3D tilt on the buttons.

- It also uses the following images:
  - *oxaca-logos.png*, as the logo of the restaurant.
  - *chef.svg*, for the staff button.
  - *customer.svg*, for the customer button.

Log-in package

The log-in package is responsible for successfully and securely connecting the staff to their equivalent user interfaces. It is implemented by the following files:

- *loginUI.html*, responsible for the format the webpage is displayed.
- *LoginStyle.css*, used to provide rules for styling the webpage.
- *LoginApp.js*, provides the functionality for securely signing-in the user.

Customer package

The customer package implements the customer ordering.

It uses the following **HTML** files:

- *menu.html,* used for formatting the menu webpage.
- *burgers.html*, used for formatting the burger submenu.
- *chicken.html*, used for formatting the chicken submenu.
- *drinks.html,* used for formatting the drinks submenu.
- *kebab.html*, used for formatting the kebab submenu.
- *meals.html*, used for formatting the meals submenu.
- *sides.html*, used for formatting the sides submenu.
- *checkout.html*, used for formatting the checkout webpage.
- *tracker.html,* used for formatting the tracking page after
a customer has checked out.

Javascript files:

- *menu.js,* provides menu functionality, including:
  - Adding to subtotal
  - Deducting from subtotal
  - Alerting the waiter
  - Popping up the allergies
  - Preps updating the order to waiting

- *checkout.js*, provides checkout functionality, including:
  - Checking out
  - Displaying a tables order
  - Viewing the order on the checkout screen.

- *menuDb.js*, is responsible for interacting with the database for menu inquiries.
  - Removing an item from an order on the database
  - Adding an item to an order on the database
  - Pushes the subtotal to the database
  - Gets the subtotal from the database
  - Contains bug fix for race condition with subtotal
  - Make new ID for random order
  - Can create a new order if needed
  - Push the new customer order back to the database

- *navbar.js,* is responsible for the movement of the side navigation bar.
  - Toggles the sidebar to in and out
  - Toggles the rest of the content to move with sidebar
- *tracker.js*, includes tilt js and handles customer enquiries after checkout
  - Offers customer to order again
  - Can request assistance

CSS files:

- *menu.css,* provides the rules for styling the menu webpage.
- *checkout.css*, provides the rules for styling the checkout webpage
- *navbar.css*, provides the rules for styling the navigation bar.
- *trackerStyle.css*, provides the rules for styling the tracker webpage

Waiter Package

The waiter package implements the user interface and functionality used by the waiter staff of the restaurant. It is composed by the following files:

- *WaiterApp.js,* responsible for waiter functionality, including:
  - Adding /Removing items from an order
  - Confirming order

  - Cancelling order

- *WaiterDb.js,* responsible for interacting with the database for waiter functionality inquiries. Such us:
  - Adding an item to an order in database
  - Removing an item from order in database
  - Deleting an order
  - Changing order status

- *WaiterTables.js*, responsible for the table management, including all interaction with the database and functionalities such as:
  - Waiter being notified for help
  - Waiter being notified that order is ready
  - Waiter being notified if a table has not yet paid
  - Tables assigned to waiter filtering

- *waiterUI.html*, implements the format of the waiter UI webpage.

- *WaiterStyle.css*, provides the rules for styling the waiter UI.

Kitchen Package

The kitchen package implements the user interface and functionality used by the kitchen staff of the restaurant. It is composed by the following files:

- *kitchenUI.html,* provides the format for displaying the kitchen UI.

- *KitchenApp.js,* implements all functionality for kitchen staff, such as:
  - Viewing Ready orders and Waiting to be prepared ones
  - Changing order status to Ready
  - Notifying waiter to deliver order

- *KitchenStyle.css*, provides the rules for styling the kitchen UI.

User Stories

<u>Completed User Stories</u>

- As a waiter, I want to be able to enter orders and produce bills.
- As a customer, I don't want my order to be lost.
- As an owner, I want a good authentication system to prevent outsiders from tampering with orders and data.
- As a customer, I want an order UI that is easy to use
- As a waiter, I want an easier way to deal with customer issues.
- As a waiter, I want an easy way to learn the menu
- As a waiter, I want to be able to easily communicate with kitchen staff

Completed User Stories - Trello

- As a customer, I want to look at the Customer Screen and browse the menu and select the desired dish. Add the selected dishes to the shopping cart and send them to the waiter interface.
- As a customer, I want to have a simple and clear way to submit an order.
- As a customer, I want to be able to have complete information on ingredients and calories for each dish so I can both avoid foods I am allergic to and follow my diet.
- As a customer I want to be able to call the waiter at any time so I can get help with my order.
- As a customer I want to see pictures of the food so that I can see what the food looks like.
- As a customer, I want to be able to track the progress of my order, so that I know how much longer I have to wait
- As a waiter, I want to be notified when a table is ready to order, so that I can confirm the order.
- As a waiter, I would like to be able to change the status of an order while customers can receive updates on their orders.
- As a waiter, I would like to be able to change the menu, so I can show customers only currently available dishes.
- As a waiter, I would like to be able to view order wait times and give priority to different orders.
- As a waiter, I want to be able to mark the order as delivered, so that the order progress is tracked correctly.
- As a waiter, I want to be assigned to a certain table, so that I won't interfere with other waiters.
- As a waiter, I want to be notified when the kitchen is ready with a dish, so that I can deliver it to the table.
- As a waiter, I would like to be able to see the payment status of each order, so I can clearly see which customers have not paid for their orders.
- As a waiter, I want to be notified when a customer needs help so I can assist them.
- As a kitchen staff, I would like to see the times the order has been waiting for on a UI.
- As a kitchen staff, I would like to be able to see when the order is placed so I know if I'm on track.
- As a kitchen staff, I want to be informed when a customer order is confirmed so I can start preparing it.
- As kitchen staff I want to be able to view all of the orders with a certain status without having to reload the page so I don't get distracted.

Video Tutorial

This is a link to a video clip presenting the restaurant management system:

https://vimeo.com/693682862